TM 851015

851Ø15 ØØ1N

NAVAL UNDERWATER SYSTEMS CENTER
NEW LONDON LABORATORY
NEW LONDON, CONNECTICUT 06320

Technical Memorandum

ORTHOGONAL POLYNOMIAL BASED ARRAY DESIGN

Date: 24 January 1985

Prepared by: Roy L. Streit

Roy L. Streit
Submarine Sonar Department

REFERENCE ONLY

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **24 JAN 1985**	2. REPORT TYPE **Technical Memo**	3. DATES COVERED **24-01-1985 to 24-01-1985**
4. TITLE AND SUBTITLE **Orthogonal Polynomial Based Array Design**		5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) **Roy Streit**		5d. PROJECT NUMBER **A70212** 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Underwater Systems Center,New London,CT,06320**		8. PERFORMING ORGANIZATION REPORT NUMBER **TM 851015**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR’S ACRONYM(S) 11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
NUWC2015

14. ABSTRACT
Array weighting designs of the Dolph-Chebyshev and Kaiser-Bessel type are based mathematically on orthogonal polynomials. The theoretical properties of these polynomials give rise to the desirable properties of the resulting arrays. This paper presents results for array weights based on a very general set of orthogonal polynomials called the Jacobi polynomials. Many interesting array far-field beampatterns are exhibited. A practical means of computing all the array weights exactly by means of one fast Fourier transform (FFT) is given. This method is quick and accurate and can compute the weights for arrays having large numbers of elements. It can efficiently compute both Dolph-Chebyshev and discrete Kaiser-Bessel weights as special cases.

15. SUBJECT TERMS
Array weighing; Dolph-Chebyshev; Kaiser-Bessel; orthogonal polynomials; Fourier transform; FFT

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**23**	

TM 851015
cy 001

NAVAL UNDERWATER SYSTEMS CENTER
NEW LONDON LABORATORY
NEW LONDON, CONNECTICUT 06320

Technical Memorandum

ORTHOGONAL POLYNOMIAL BASED ARRAY DESIGN

Date: 24 January 1985

Prepared by: Roy L. Streit

Roy L. Streit
Submarine Sonar Department

ABSTRACT

Array weighting designs of the Dolph-Chebyshev and Kaiser-Bessel type are based mathematically on orthogonal polynomials. The theoretical properties of these polynomials give rise to the desirable properties of the resulting arrays. This paper presents results for array weights based on a very general set of orthogonal polynomials called the Jacobi polynomials. Many interesting array far-field beampatterns are exhibited. A practical means of computing all the array weights exactly by means of one fast Fourier transform (FFT) is given. This method is quick and accurate and can compute the weights for arrays having large numbers of elements. It can efficiently compute both Dolph-Chebyshev and discrete Kaiser-Bessel weights as special cases.

ADMINISTRATIVE INFORMATION

This technical memorandum was prepared under NUSC IR/IED Project No. A70212, "Jacobi Digital Filter and Antenna Design," Principal Investigator, R. L. Streit (Code 3232). The Program Manager is CAPT. Z. L. Newcomb, Chief of Naval Material (MAT 05B).

I. INTRODUCTION

Array weights of the Dolph-Chebyshev and Kaiser-Bessel type are based on orthogonal polynomials. The desirable properties of these arrays are due entirely to the properties of the underlying orthogonal polynomials. In this paper the mathematical design methodology developed in [1], which parallels the techniques of the Dolph-Chebyshev designs, is further explored for Jacobi polynomials.

Analytical expressions for the underlying weights are not sought here, as they are in [1], because such expressions are probably not competitive with the exact FFT based method presented in this paper. The primary purpose of this paper is to present a unified FFT based method for computing array weights based on any of the Jacobi polynomials. This method is quick and accurate and can compute the weights for arrays having large numbers of elements. The Appendix gives a short Fortran program for computing the weights (given a subroutine for the FFT). This program can efficiently compute both Dolph-Chebyshev and discrete Kaiser-Bessel weights as special cases.

This paper represents, in a sense, a completion of certain ideas about array design using orthogonal polynomials. Dolph was apparently the first to use an orthogonal polynomial for designing an array weighting function. The polynomial he used was the Chebyshev polynomial of the first kind, $T_n(x)$, and he was able to prove an optimality condition. Unfortunately his proof of this optimality condition relies on the unique behavior of the graph of $T_n(x)$, and so generalizations of the optimality condition seem unlikely. Nonetheless, the use of different orthogonal polynomials can lead to useful weighting functions. For example, if the Chebyshev polynomial of the second kind, $U_n(x)$, is used in place of $T_n(x)$ a weighting function of Kaiser-Bessel type results (see [1]). It is also possible to use a general family of orthogonal polynomials that contains both $T_n(x)$ and $U_n(x)$ as special cases. The Gegenbauer polynomials, $C_n^{\mu}(x)$, are one such family; that is, $T_n(x)$ and $U_n(x)$ are the special cases $\mu = 0$ and $\mu = 1$, respectively. This family is used in [1] and gives useful and interesting designs. The most general of the so-called classical orthogonal polynominals that contains all these examples as special cases is the family of Jacobi polynomials, $P_n^{(\alpha,\beta)}(x)$. For $\alpha = \beta = \mu - 1/2$ they reduce to $C_n^{\mu}(x)$. Do Jacobi polynomials turn out to be useful? The examples presented in this paper indicate that, although many interesting new array designs are possible using the Jacobi polynominals, the most useful designs in this general family are probably those that have already been discussed. Consequently the new Jacobi designs might be said to be, at present, "a solution looking for a problem."

II. WEIGHT GENERATION BY FFT

The far-field beampattern of a general linear beamformer for a linear equispaced array having 2N + 1 elements ($N \geq 1$) with element positions $x_k = k\,\lambda/(\nu D)$, $k = 0, \pm 1, \ldots, \pm N$, is given by:

$$F(u) = \sum_{k=-N}^{N} w_k \exp(-i2\pi\, ku/(\nu D)) \tag{1}$$

where the integer $D \geq 1$ is given and $\nu > 0$ is a fixed real constant (νD is the number of elements per wavelength), λ is the wavelength of the design frequency, and u is defined by

$$u = \sin \theta_a - \sin \theta_s, \tag{2}$$

where θ_s, $-\pi/2 \leq \theta_s \leq \pi/2$, is the steering (look) angle and θ_a, $-\pi/2 \leq \theta_a \leq \pi/2$, is the arrival angle of a plane wave. Both angles are measured from a line normal to the array axis. The weights $\{w_k\}$ can be, in general, any set of complex constants.

Define the functions

$$t_D(z) = \sum_{k=-D}^{D} a_k z^k \tag{3}$$

$$P_n(z) = \sum_{k=0}^{n} b_k z^k \quad , n \geq 0, \tag{4}$$

where $\{a_k\}$ and $\{b_k\}$ are specified constants. By simple algebra

$$P_n(t_D(z)) = \sum_{k=-nD}^{nD} c_k z^k \tag{5}$$

where $\{c_k\}$ depend on both $\{a_k\}$ and $\{b_k\}$. Substituting $z = \exp(-i\pi u/(\nu D))$ gives

$$H(u) = P_n(t_D(\exp(-i\pi u/(\nu D)))) \tag{6}$$

$$= \sum_{k=-nD}^{nD} c_k \exp(-i2\pi ku/(2\nu D)). \tag{7}$$

By comparing (7) with (1), it is seen that $H(u)$ is the far-field beampattern of a linear array with $2nD + 1$ elements, equispaced $\lambda/(2\nu D)$ apart, and with the weight c_k applied to the k-th element. It will be shown that the array weights c_k can be computed from function values of t_D and P_n by means of an FFT.

When some of the weights $c_k = 0$, the corresponding elements may be eliminated from the physical array without altering the array's far-field beampattern. Elimination of zero weighted elements (whenever possible) minimizes the total number of elements required. This consideration is especially important when t_D and P_n are chosen so that $P_n(t_D(z))$ is either even or odd in z, for then about half the elements need not be physically present. This is discussed further in section III.

We now show that the weights $\{c_k\}$ in (7) can be computed exactly from (3) and (4) by the fast Fourier transform (FFT). It is stressed that this procedure is theoretically <u>exact</u>, not approximate, for the $\{c_k\}$.

Let $f(u)$ be any complex valued function of a real variable u which can be written exactly in the form

$$f(u) = \sum_{k=-p}^{p} d_k \, e^{-iku} \tag{8}$$

for some complex constants $\{d_k\}$. Let

$$f_k = \frac{1}{2p} \sum_{j=0}^{2p-1} F_j \, e^{i2\pi kj/(2p)}, \quad k = 0, 1, \ldots, 2p-1, \tag{9}$$

where

$$F_j = (-1)^j f((p-j)\pi/p), \quad j = 0, 1, \ldots, 2p-1. \tag{10}$$

Thus $\{f_k\}$ is the inverse FFT of order 2p of the sequence $\{F_j\}$. Substitute (8) into (10), and then into (9) to get

$$f_k = \frac{1}{2p} \sum_{j=0}^{2p-1} \left\{ (-1)^j \sum_{q=-p}^{p} d_q \, e^{-i\pi q(p-j)/p} \right\} e^{i\pi kj/p}$$

$$= \frac{1}{2p} \sum_{q=-p}^{p} (-1)^q d_q \cdot \left\{ \sum_{j=0}^{2p-1} (-1)^j e^{i\pi(q+k)j/p} \right\}, \quad k=0,1,\ldots,2p-1.$$

The inner sum equals 2p when $q + k = \pm p, \pm 3p, \ldots$ and equals zero otherwise. Since $d_q = 0$ for $|q| > p$,

$$f_k = \begin{cases} (-1)^p \, (d_{-p} + d_p), & \text{if } k = 0 \qquad (11.a) \\ (-1)^{p-k} \, d_{p-k}, & k = 1, \ldots, 2p-1. \qquad (11.b) \end{cases}$$

Now let $f(u) = H(\nu D u/\pi)$, where $H(u)$ is given by (7), and $p = nD$. Then

$$F_j = (-1)^j \, P_n(t_D(e^{-i\pi(nD - j)/nD})), \; j = 0, 1, \ldots, 2nD-1,$$

and the coefficients c_k in (7) are just

$$c_k = \begin{cases} a^n_{-D} \, b_n & \text{, if } k = -nD, \qquad (12.a) \\ (-1)^{k-nD} \quad f_{nD-k} & \text{, if } k = -nD+1, \ldots, nD-1 \qquad (12.b) \\ a^n_D \, b_n & \text{, if } k = nD, \qquad (12.c) \end{cases}$$

where $\{f_k\}_0^{2nD-1}$ is the inverse FFT of $\{F_j\}_0^{2nD-1}$. The coefficients c_{-nD} and c_{nD} cannot be computed directly by FFT because of aliasing, as indicated in (11.a); however, by direct appeal to the defining equations (3) - (6), it follows that c_{-nD} and c_{nD} are as given by (12.a) and (12.c), respectively. In fact, (11.a) can be used as a check on numerical accuracy in the computations.

It should be obvious that some special forms of t_D and P_n can be used advantageously to reduce the size of the FFT required to compute (12). In general, however, no special structure exists and the smallest FFT size that can be used has order 2nD.

In cases where 2nD is not an integer power of two, the FFT is still applicable by zero filling in t_D and P_n. That is, D is replaced by the smallest power of two which exceeds or equals D, say D'. The function t_D is then merely considered to be a special case of $t_{D'}$. Similarly, P_n is a special case of $P_{n'}$ for some smallest power of two, n', which is greater than or equal to n. The required size of the FFT is thus 2n'D'. The coefficients $\{c_k\}$ are still given by (12); however, from (7), it must be the case that $c_k = 0$ for $|k| > nD$.

The Appendix gives a Fortran subroutine for computing the array weights by an FFT, given subroutines for evaluating $P_n(z)$ and $t_D(z)$. The program is specialized to the case D = 1, but it can be easily altered to accommodate larger values of D. The program is also written on the assumption that 2nD is a power of two. As discussed in the preceding paragraph, zero filling allows the most general situation to go through.

III. THE TEN PARAMETER JACOBI WEIGHT FAMILY

The most important special case of $H(u)$ seems to be $D = 1$. Although there may be interesting possibilities when $D > 1$ (consider, for example, the identity $T_n(T_D(\cos\theta)) = T_{n+D}(\cos\theta)$, where T_n is the Chebyshev polynomial of the first kind), these cases are not explored in this paper.

Before proceeding, it is instructive to see first how the usual Dolph-Chebyshev case for half wavelength equispaced arrays is derived as a special case of $H(u)$. Let

$\tilde{N}$ = number of array elements
$D = 1$
$\nu = 2$
$\tilde{n} = \tilde{N} - 1$
$t_1(z) = Z_0(z^{-1} + z)/2$
$P_{\tilde{n}}(z) = T_{\tilde{n}}(z)$,

where $T_{\tilde{n}}(z)$ is the Chebyshev polynomial of the first kind and the real constant Z_0 is given by

$$Z_0 = \frac{1}{2}\left\{(Q + (Q^2-1)^{1/2})^{1/\tilde{n}} + (Q - (Q^2-1)^{1/2})^{1/\tilde{n}}\right\} \tag{15}$$

where

$Q = 10^{|S|/20}$
S = specified sidelobe level (in dB).

For these values it follows that $t_1(\exp(-i\pi u/2)) = Z_0\cos(\pi u/2)$ and, from (6) and (7),

$$H(u) = T_{\tilde{n}}(\, Z_0 \cos(\pi u/c)) \tag{16}$$

$$= \sum_{k=-\tilde{n}}^{\tilde{n}} c_k \exp(-i\pi k u/2). \tag{17}$$

By comparing (17) to (1), it would appear at first glance that this array is <u>quarter</u> wavelength equispaced with $2\tilde{n} + 1 = 2\tilde{N} - 1$ elements. However, every other coefficient in (17) is identically zero because $T_{\tilde{n}}(z)$ is always either an even or an odd function in z. Deleting the zero weighted elements reduces (17) to two slightly different cases, depending only on whether $\tilde{N}$ is even or odd. These cases are not given explicitly here. Note, however, that $\tilde{N}$ even implies that $\tilde{n}$ is odd and that $T_{\tilde{n}}(z)$ contains no even powers of z, which means that $T_{\tilde{n}}(z)$ has $(\tilde{n} + 1)/2$ non-zero coefficients and, consequently, that $T_{\tilde{n}}(t_1(z))$ has precisely $\tilde{n} + 1 = \tilde{N}$ non-zero terms in the expansion (17). Similar reasoning holds for $\tilde{N}$ odd.

To summarize: The Dolph-Chebyshev array design is thought of, in the context of this paper, as a quarter wavelength equispaced array in which half the elements (every other one) has been zero weighted. Dolph-Chebyshev arrays of both even and odd numbers of elements are thought of in this way.

From (3), the most general form for D = 1 is

$$t_1(z) = a_{-1}\, z^{-1} + a_0 + a_1\, z.$$

For reasons that will become clear, the slightly more restrictive form

$$t_1(z) = z_0(r_0^{-1}\, z^{-1} + a_0 + r_0\, z)/2,\ r \neq 0, \qquad (18)$$

is adopted, where z_0, r_0, and a_0 are arbitrary complex constants. The only useful form excluded by (18) is obtained by changing the sign of the highest order term; this latter form corresponds to "difference" patterns and, for ease of exposition, is not discussed further. For the remainder of the paper, (18) is taken as the definition of $t_1(z)$. Note that $t_1(e^{-i\theta}) = z_0 \cos\theta$ if $r_0 = 1$ and $a_0 = 0$.

The most general class of polynomials $P_n(z)$ considered in this paper are the Jacobi polynomials, denoted $P_n^{(\alpha,\beta)}(z)$. They are defined explicitly by

$$P_n^{(\alpha,\beta)}(z) = \frac{1}{n!} \sum_{k=0}^{n} \binom{n}{k} (n+\alpha+\beta+1)_k\ (\alpha+k+1)_{n-k} \left(\frac{z-1}{2}\right)^k \qquad (19)$$

for all complex values of α, β, and z. For $\alpha > -1$ and $\beta > -1$, the Jacobi polynomials are orthogonal on the real interval $-1 \leq z \leq +1$, but they are not necessarily orthogonal for other values of α and β. The best available method for computing $P_n^{(\alpha,\beta)}(z)$ relies not on (19) but on the three term recursion which they satisfy. The published algorithm [2], [3] based on this recursion is easily modified to compute the Jacobi polynomials for complex α, β, and z for all values of the degree n that are likely to be of practical interest, say $n \leq 150$. A thorough mathematical treatment of $P_n^{(\alpha,\beta)}(z)$ is available in [4].

The generalized Laguerre and Hermite polynomials, together with the Jacobi polynomials, constitute a complete list of the so-called classical orthogonal polynomials. Array designs can also be based on the Laguerre and Hermite polynomials and will, of course, be different from those based on Jacobi's. Although these designs are probably interesting, in this paper attention is restricted to the Jacobi polynomials.

The use of (18) and (19) gives rise to a five parameter family of weights which includes nearly all the well known analytic families of weights as special cases. (The most prominent exception is Taylor weighting). The five parameters are z_0, r_0, a_0, α, and β. Each parameter can be complex, so there are actually 10 real parameters if the real and imaginary parts of each are counted separately. The Fortran program listed in the Appendix is written for this general case.

The simplest way to explore the properties of these ten parameters is to perturb each parameter separately while holding the others fixed at some nominal value. The nominal parameter values chosen here for the examples are those that give rise to the Dolph-Chebyshev design for an array of 33 elements with a sidelobe level of -30 dB. Specifically,

$$\begin{aligned} \alpha_0 &= -.50 + 0.0\,i \\ \beta_0 &= -.50 + 0.0\,i \\ a_0 &= 0.0 + 0.0\,i \\ r_0 &= 1.0 + 0.0\,i \\ z_0 &= Z_0 = 1.008408 + 0.0\,i, \end{aligned} \tag{20}$$

where Z_0 is computed from (15) with $S = -30$ dB and $\tilde{n} = 32$. Recalling earlier remarks in this section concerning the interpretation of half wavelength equispaced arrays as quarterwavelength equispaced arrays with zero weights, set $n = 32$ in (6) and (7). Thus, in principle, the example is a $2n + 1 = 65$ element quarterwavelength equispaced array. The coefficients $\{c_k\}$ are computed from (12). In (12.a) and (12.c), note that the identity [4, Eq. (4.21.6)]

$$b_n = 2^{-n} \binom{2n + \alpha + \beta}{n}, \quad n \geq 0, \tag{21}$$

holds and so, from (18),

$$a_D = \frac{1}{2} z_0 r_0, \quad a_{-D} = \frac{1}{2} z_0 r_0^{-1} \tag{22}$$

Each of the ten parameters is both increased as well as decreased from its nominal value given in (20). Thus there are 21 cases, including the nominal case itself in (20). Table 1 displays these cases and gives each an identifying case name. To each case in Table 1 there corresponds a graph of the far-field beampattern H(u) on the u-interval [0,4] and two bar charts, one for the real part and one for the imaginary part of the weights corresponding to that case.

Figure Number	Case Names				Value of Perturbed Parameter			
1	NOM				no deviations from (20)			
2	Z.1	Z.2	Z.3	Z.4	z_0+.003	z_0-.003	z_0+.003i	z_0-.003i
3	A.1	A.2	A.3	A.4	a_0+.003	a_0-.003	a_0+.003i	a_0-.003i
4	R.1	R.2	R.3	R.4	r_0+.03	r_0-.03	r_0+.03i	r_0-.03i
5	α.1	α.2	α.3	α.4	α_0+.3	α_0-.3	α_0+.3i	α_0-.3i
6	β.1	β.2	β.3	β.4	β_0+.3	β_0-.3	β_0+.3i	β_0-.3i

Table 1 Perturbed parameter values; deviation from the nominal values (20)

In general, H(u) is periodic with a period of length $2\nu D$. Since $D = 1$ and $\nu = 2$ in these examples, any interval of length 4 suffices to exhibit all the structure of H(u).

In all the bar charts presented, upward lines indicate positive weights and length is proportional to magnitude. Similarly, downward lines indicate negative weights. These upward and downward lines are ordered from left to right and correspond to elements numbered from -32 to +32. Any element receiving a zero weight is indicated by a simple "x" marking its position. In particular, notice that the nominal case, NOM, of Figure 1 has only 33 non-zero weights. The nominal case, as has been said, is a half wavelength equispaced array being treated as a quarter wavelength equispaced array. The weights in each case are normalized by the largest magnitude of any real or imaginary part; thus, the normalization between cases is not exactly the same.

Figure 1 is the reference case (20) and needs no further comment.

Figure 2 perturbs only z_0. The cases Z.1 and Z.2 are expected since z_0 is merely increased or decreased in its real part alone. When z_0 is perturbed by adding an imaginary component, the array still has 33 non-zero weights and so is, in effect, half wavelength equispaced. It is surprising how much can be added to the imaginary parts of the weights without seriously degrading the beampattern. The beampatterns in Z.3 and Z.4 are identical.

Figure 3 perturbs a_0 from its nominal zero value. Any perturbation produces a quarter wavelength equispaced array which is symmetrically weighted. One way to discuss the results is to visualize the 65 element array as being composed of two half wavelength equispaced arrays---one having 33 elements and the other 32 elements with the elements of the two arrays interlaced. Thus, perturbing the real part of a_0 is equivalent to adding or subtracting the outputs of these two arrays. Perturbing the imaginary part of a_0 is equivalent to adding or subtracting the outputs after first putting them in phase quadrature with respect to each other. The beampatterns A.3 and A.4 are identical to each other, but they are NOT the same as Z.3 and Z.4.

Figure 4 perturbs r_0 from its nominal value of +1. Any small perturbation produces asymmetrically weighted half wavelength equispaced arrays. Real perturbations of r_0 produce only real weights and have beampatterns without any true nulls. Pure imaginary perturbations do not alter the beampattern from its nominal case, even though the weights develop an interesting sinusoidal character in their imaginary parts.

Figure 5 perturbs α from its nominal value of $\alpha_0 = -1/2$. Any perturbation produces a quarter wavelength equispaced array which is symmetrically weighted. Real perturbations yield real weights while pure imaginary perturbations yield complex weights. The first grating lobe in case α.1 is at about -8 dB instead of 0 dB; the same is true of the MRA in case α.2.

Figure 6 perturbs β from its nominal value of $\beta_0 = -1/2$. Any perturbation produces quarter wavelength equispaced arrays which are symmetrically weighted. Real perturbations yield real weights, while pure imaginary perturbations yield complex weights. The first grating lobe in case β.2 is suppressed to about -8 dB, while in case β.1 the MRA is depressed to -8 dB. Figure 6 and Figure 5 should be compared.

The following observations seem to hold:

1. The real part of α controls the "upper" envelope of $H(u)$ near the center peak.

2. The absolute value of the imaginary part of α controls the "lower" envelope of $H(u)$ near the center peak.

3. The real part of β controls the "upper" envelope of $H(u)$ near the first grating lobe.

4. The absolute value of the imaginary part of β controls the "lower" envelope of $H(u)$ near the first grating lobe.

Other parameters (the imaginary parts of a and z_0) also affect the "lower" envelope of $H(u)$, but the dominant effects seem to be due to the imaginary parts of α and β. The imaginary part of r does not affect the lower envelope at all.

By changing the parameters simultaneously in different ways, the different effects may be combined, at least for small perturbations. Examples of this are not included here.

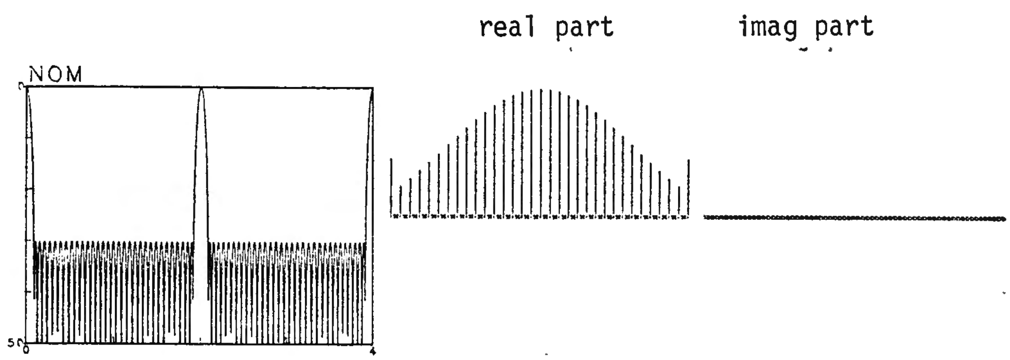

Figure 1. No perturbations; the nominal case (20)

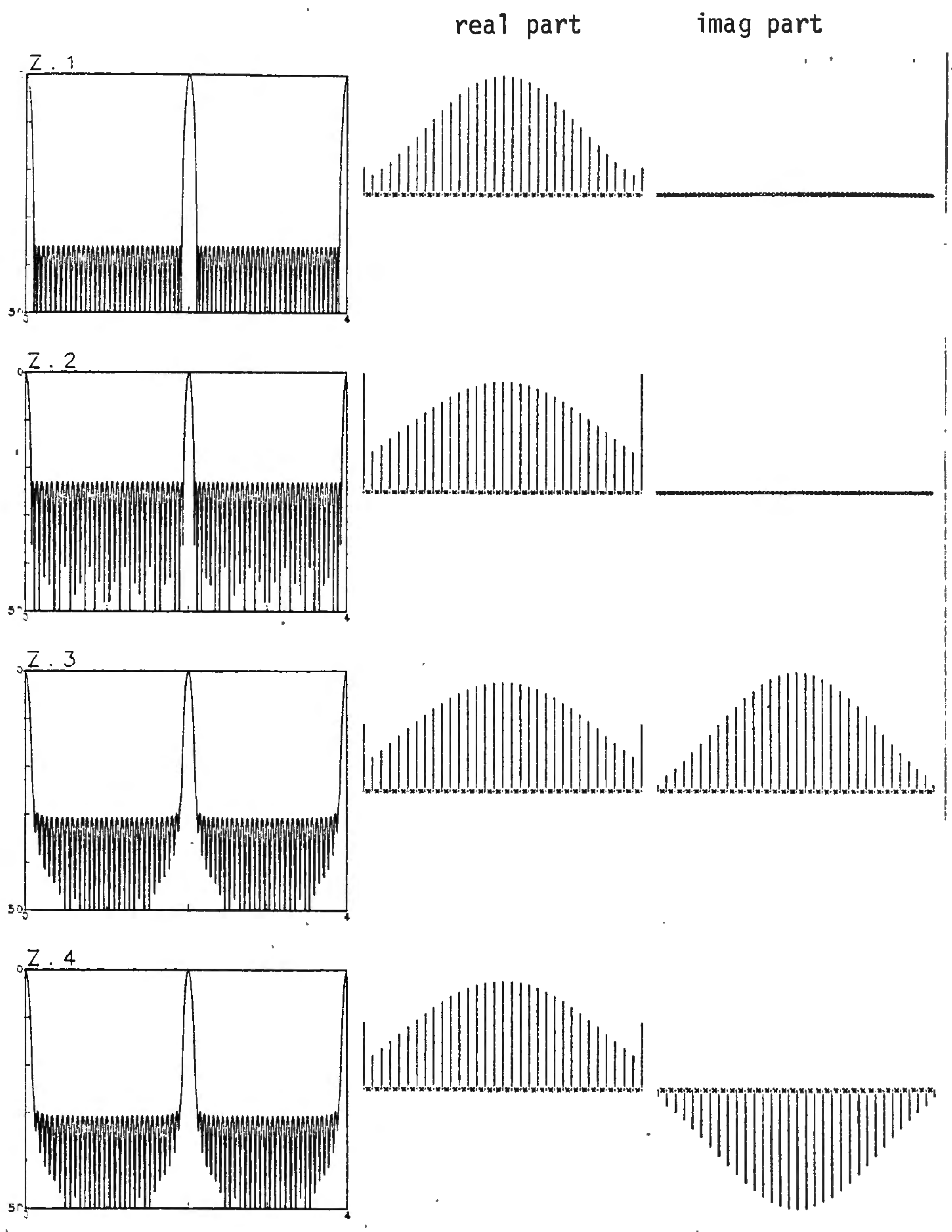

Figure 2. Perturbations of z_0

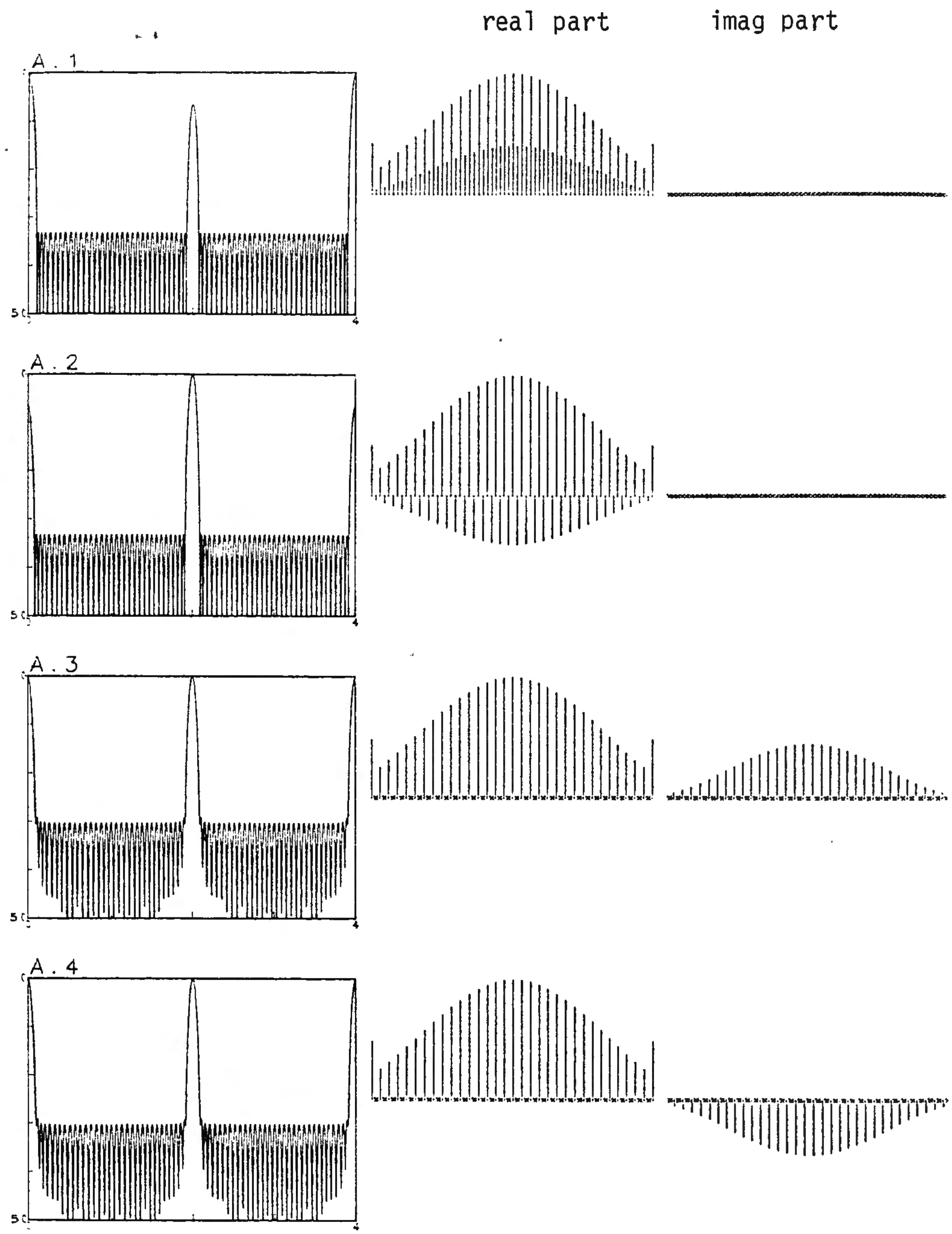

Figure 3. Perturbations of a_0

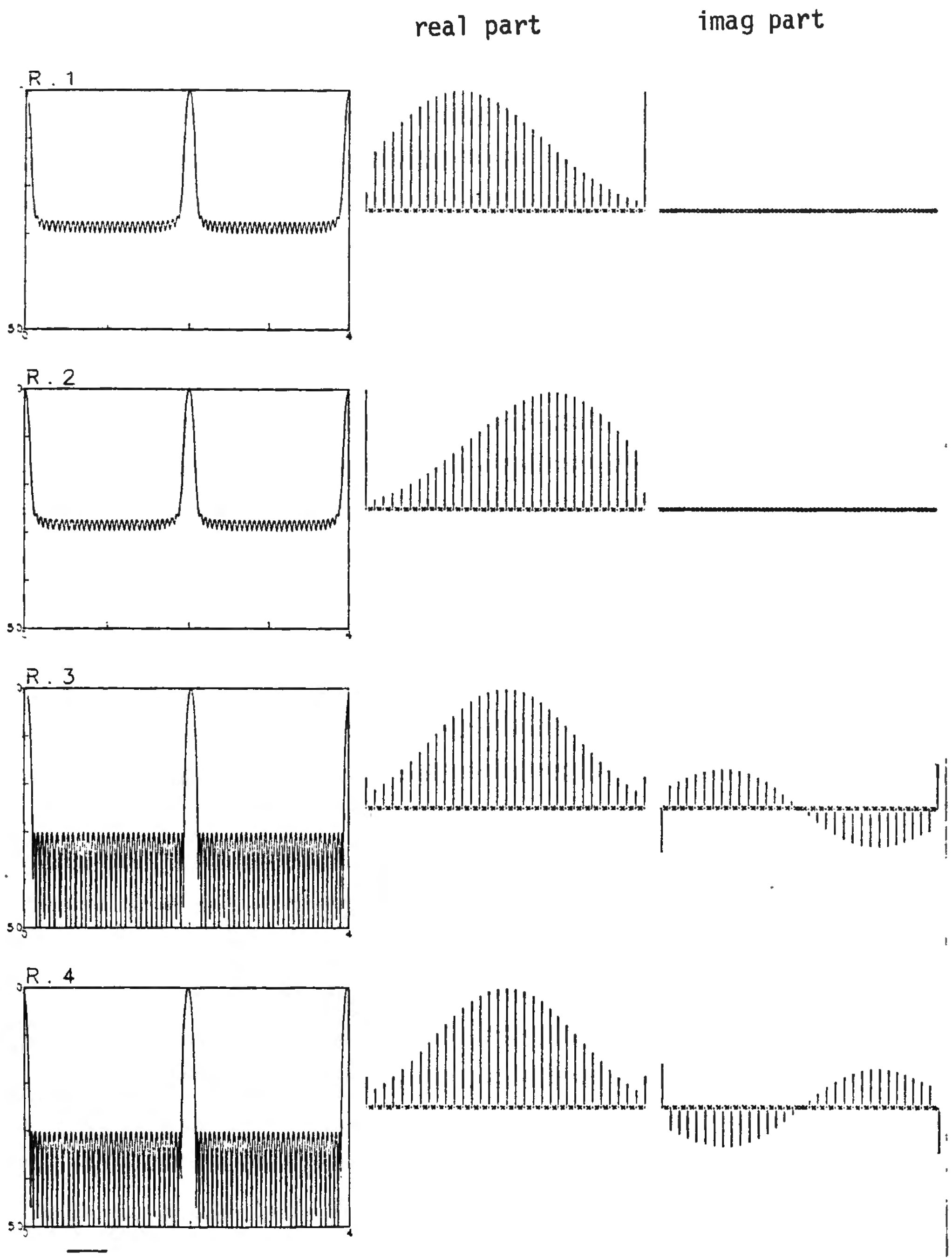

Figure 4. Perturbations of r_o

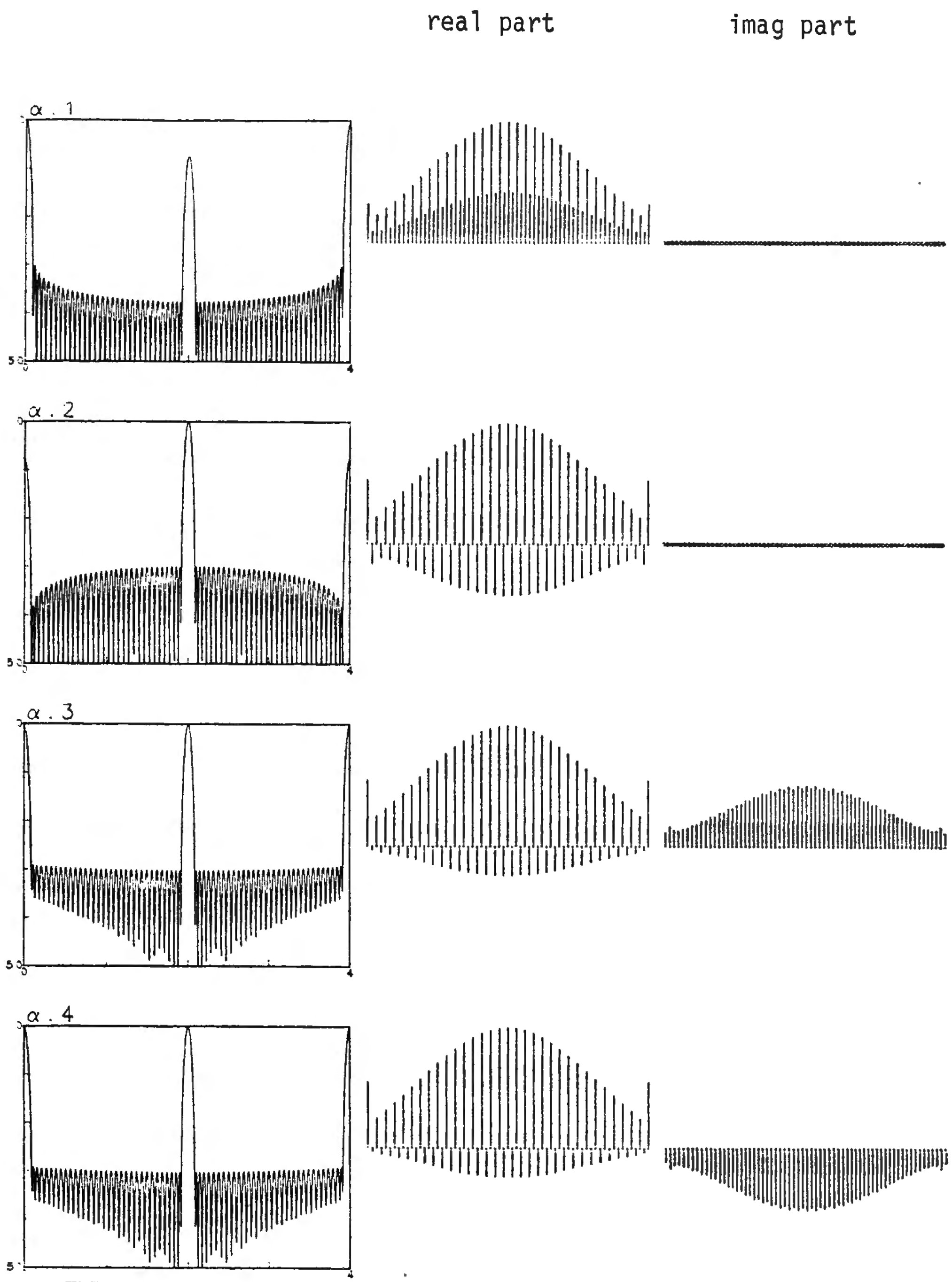

Figure 5. Perturbations of α_0

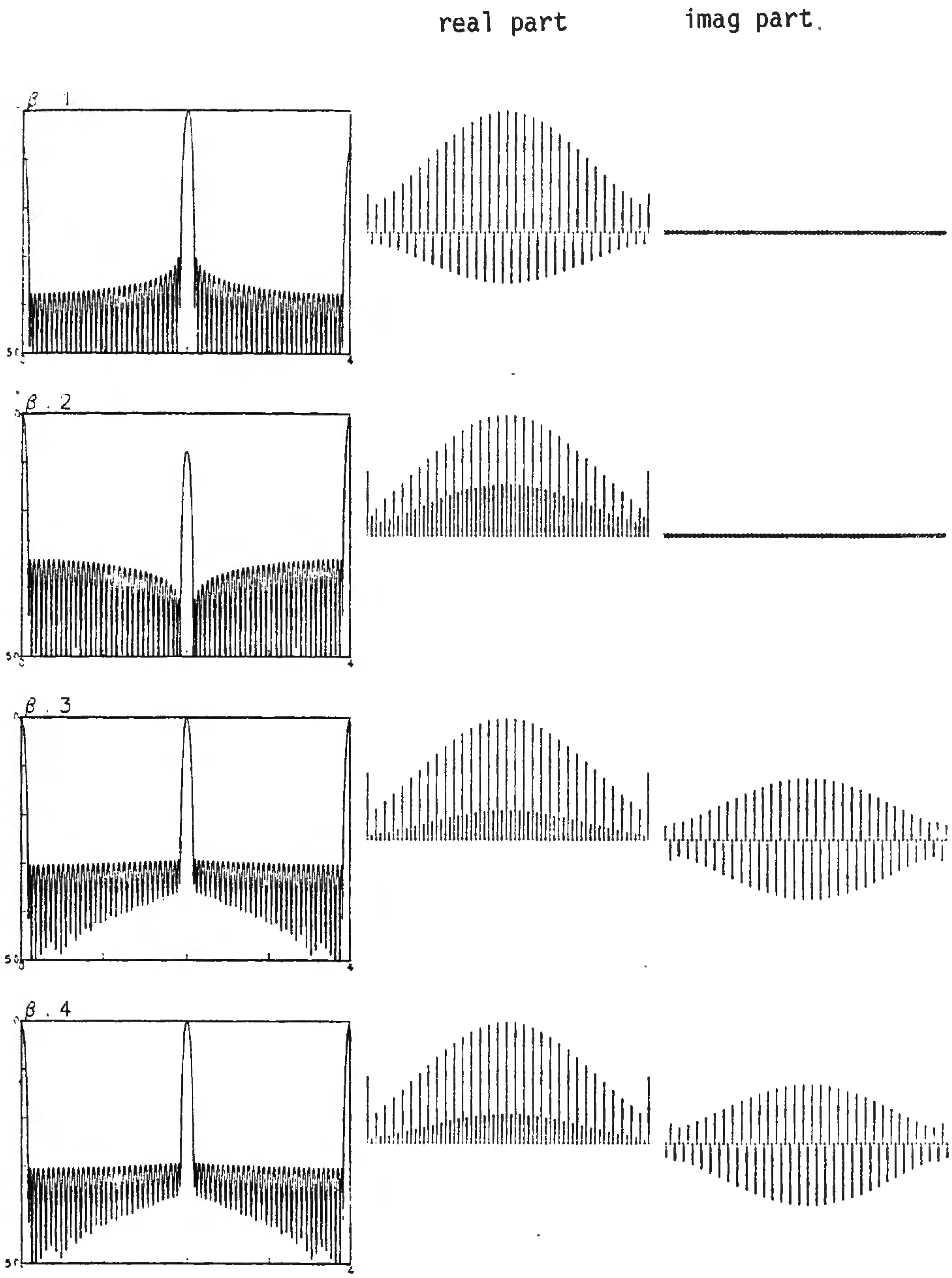

Figure 6. Perturbations of β_0

IV. SUMMARY AND CONCLUDING REMARKS

It has been shown that array weights based on the Jacobi orthogonal polynomials can be computed exactly by means of FFT. As a special case, weights based on the Gegenbauer polynomials can also be computed exactly by FFT, instead of analytically as in [1]. Examples have been presented to show the effects of varying the ten parameters in the Jacobi family.

Further work in this area is possible. In addition to the Jacobi polynomials, one may also use the generalized Laguerre and the Hermite polynomials. In fact, any orthogonal polynomial family that has interesting structural features can be the basis of a weighting family which inherits this structure. In a different direction, certain cases for $D > 1$ may yield interesting designs and have not been explored. The weights corresponding to all these cases can computed exactly by the FFT method presented in this paper.

APPENDIX

The Fortran program JACWTS listed below assumes that D = 1 and that 2nD is a power of 2. The function $t_D(z)$ is defined exactly as in (18), and the polynomial $P_n(z)$ is taken to be the Jacobi polynomial $P_n^{(\alpha,\beta)}(z)$.

This program is an implementation of the (exact) FFT method described by Eq. (12), where $a_{\pm D}$ and b_n are given by (22) and (21), respectively. The user of JACWTS need only specify values for α, β, a_0, r_0, and z_0. In JACWTS these variables are referred to by the labels ALPHA, BETA, AO, RO, ZO, respectively. The arrays X and Y contain, on output, the real and imaginary parts of the array weights $\{c_k\}$. These two arrays must be dimensioned at least 2nD+1 in the routine which calls JACWTS. The integers n and D are referred to by the labels N and D, respectively, in the subroutine argument list. Also, LOGN and LOGD are defined so that N = 2**LOGN and D = 2**LOGD.

This program assumes that a subroutine named JACOBI evaluates the Jacobi polynomial (19) for arbitrary complex values of α, β, and z. This subroutine can be based on the published codes in [2] and [3]. This program also assumes that subroutines are available for computing a complex FFT of size 2nD; the particular ones used here are based on Markel's method and are not listed. Their names are DPMCOS and DPMFFT. These routines require a work array, C, dimensioned at least 2nD in the routine which calls JACWTS.

JACWTS is written in double precision complex mode to forestall any numerical round-off error problems that might arise. The test suggested in Section II (that follows from the resolution of the aliasing effects as in (12)) is incorporated. It is the only test used to ascertain whether numerical round-off of significant proportions occurred. No numerical difficulties have been detected by this test to date, which indicates that the computation is usually numerically reliable.

```
      SUBROUTINE JACWTS(X,Y,C,N,LOGN,D,LOGD,ALPHA,BETA,A0,R0,Z0)
      COMPLEX*16 Z,T,H,S,R,TSUBD,ALPHA,BETA,Z0,A0,R0,JACOBI
      INTEGER N,LOGN,D,LOGD,TWOND
      DOUBLE PRECISION EPSI,ARG,PI,X(1),Y(1),C(1)
      DATA PI,EPSI/3.14159265358979323800,0.5D-7/
      M=LOGN+LOGD+1
      ND=N*D
      TWOND=2*ND
C     ... ALL WEIGHTS EXCEPT THE FIRST AND THE LAST
      I=+1
      DO 10 J=1,TWOND
      ARG=-PI*(ND-J+1)/ND
      Z=DCMPLX(COS(ARG),SIN(ARG))
      T=TSUBD(D,A0,R0,Z0,Z)
      H=JACOBI(N,ALPHA,BETA,T)
      X(J)=I*DREAL(H)
      Y(J)=I*DIMAG(H)
      I=-I
   10 CONTINUE
      CALL DPMCOS(C,TWOND)
      CALL DPMFFT(X,Y,C,M,+1)
      I=+1
      DO 15 J=1,TWOND
      X(J)=I*X(J)/TWOND
      Y(J)=I*Y(J)/TWOND
      I=-I
   15 CONTINUE
      ... THE FIRST AND LAST WEIGHTS
      H=.25D0*Z0*R0
      S=.25D0*Z0/R0
      R=1.0D0
      T=1.0D0
      DO 18 I=1,N
      Z=((N+N-I+1)+ALPHA+BETA)/I
      T=T*H*Z
      R=R*S*Z
   18 CONTINUE
      X(1)=DREAL(R)
      Y(1)=DIMAG(R)
      X(TWOND+1)=DREAL(T)
      Y(TWOND+1)=DIMAG(T)
      ... NUMERICAL ACCURACY TEST
     ARG=(ABS(X(1)-DREAL(R+T))+ABS(Y(1)-DIMAG(R+T)))
    +      /(1.0D0+ABS(X(1))+ABS(Y(1)))
      IF(ARG.GT.EPSI)PRINT 50
   50 FORMAT(' NUMERICAL ROUND-OFF ERROR IS SIGNIFICANT.')
      RETURN
      END
      FUNCTION TSUBD(D,A0,R0,Z0,Z)
      COMPLEX*16 Z,Z0,A0,R0,TSUBD
      INTEGER D
C     TREAT THE CASE D=1; IGNORE OTHER VALUES.
      TSUBD=.5D0*Z0*( (1.0D0/(R0*Z)) + A0 + (R0*Z) )
      RETURN
      END
```

REFERENCES

1. R. L. Streit, "A Two-Parameter Family of Weights for Nonrecursive Digital Filters and Antennas," IEEE Trans. on ASSP, Vol. ASSP-32, February 1984, pp. 108-118.

2. B. F. W. Witte, "Algorithm 332, Jacobi Polynomials," Comm. ACM, Vol. 11, June 1968, p. 436.

3. O. Skovgaard, "Remark on Algorithm 332," Comm. ACM, Vol. 18, February 1975, pp. 116-117.

4. G. Szego, Orthogonal Polynomials, Fourth Edition, AMS Colloquium Publications, Vol. XXIII, American Mathematical Society, Providence, RI, 1975.

External Distribution

NAVSEA, PMS411C, W. Burgess
PMS409, L. McGonegle
L. Kneesi
Code 63R-1
Code 63-R12, C. Walker
Code 63X1, G. McBurnett

NAVELEX, Code ELEX612, T. Higbee

NOSC, Code 7122, S. Nichols

DTNSRDC, Bethesda, MD, Code 19832, G. Mabry

Bell Laboratories, (Contract # N00024-83-C-6003)
G. Zipfeil, Jr.

Bendix, Co., Sylmar, CA (Contract #N00140-82-G-BZ16-0004)
F. Demetz, Sr.

Gould, Glen Burnie, (Contract # NICRAD-84-NUSC-022)
F. Lehman
C. Kennedy

Hughes Aircraft, Fullerton
(Contract # NICRAD-83-NUSC-001B)
D. Veronda

Technology Service Corporation, Salida, CO
(Contract # N00140-82-G-BZ44-0002)
L. Brooks